Parallelism Drives Performance: A Perspective on the Future of Processing

Q&A with Intel Fellow David J. Kuck

This article originally appeared in the Intel® Software Insight Magazine - July 2006.

Table of Contents

(Click on page number to jump to sections)

Parallelism Drives Performance: A Perspective on the Future of Processing

This article originally appeared in the Intel® Software Insight Magazine - July 2006.

Overview: The Promise of Massively Parallel Computing

Look at the latest series of Intel technology roadmaps and the direction becomes clear: multi-core processing is taking the industry on a fast-moving and exciting ride into profoundly new territory. The defining paradigm in computing performance has shifted inexorably from raw clock speed to parallel operations and energy efficiency. Industry participants are moving swiftly to adapt to the change. Noted Intel Fellow, David J. Kuck, who has been working for several decades in this area, describes the promise of multi-core processing and massively parallel computing as well as the challenges presented to the computer industry.

> "Across the board, Intel is supporting undergraduate education, providing a strong selection of very advanced tools, and engaging in direct enabling activities with ISVs."
>
> —David Kuck, Intel Fellow, Software and Solutions Group

What experience do you have in parallel computing?

I have actually been working on parallel computing for 40 years. I first started in this area as a professor—we built parallel machines at universities, including the University of Illinois at Urbana-Champaign, and I also did a lot of consulting in the '70s and '80s. Then, we started producing software through my little company—KAI—in 1979. The concepts have been around for a very long time. There has been a trickle down from supercomputers to mini-supercomputers and servers that has now reached Intel chips—the multi-core processors.

At the same time, the applications have gone from the very highest end of technical computing to more about throughput and desktop applications. Until recently, there hasn't been an economic incentive to push parallel computing. Now we are at the crossroads, and the economic incentives are enormous. A long list of research ideas and advanced development ideas and products are coming to the fore.

Multithreading Produces Gains in Application Behavior

Is it accurate to say that in the future, performance enhancements will increasingly depend on multithreading and that will be the most effective way to produce gains in application behavior?

It is a necessity that the software developers who want to take advantage of multi-core architectures learn how to move beyond two or four cores to making effective use of eight, sixteen, or more. With two cores, you simply have tasks scheduled effectively by the operating system. For example, you can have a background virus checker and a foreground Windows* application—with one running on each core. No problems in terms of cooling off the cache or switching contexts—you can run them at the same time.

With eight or sixteen cores, the situation is different. When we look at the desktop environment, if you have Windows open, there are hundreds of threads active. But they're not really doing anything, and there is not the potential for them to do a whole lot concurrently unless something in the programming model changes. If we want a game engine, for example, to take advantage of eight or sixteen cores, someone needs to devote some energy to rewriting that game engine.

What advances are we seeing in tools to enhance multithreaded application development?
There are advances in a number of areas, and Intel is active in all of them. Languages better suited to multithreaded development are a growing area of interest—we have been pioneers and leaders in OpenMP*, and we are now extending the benefits of OpenMP to C++ developers with a new scalable parallel programming library that will be made publicly available later this year.

Tools designed to robustly support threading techniques and help ensure correctness and performance are another area where progress is being made. The Intel® Threading Tools support OpenMP and threading models with Intel® Thread Profiler and Intel® VTune™ Performance Analyzer, as well as our innovative Intel® Thread Checker that can find many common parallelism errors automatically.

Richer software libraries with features that support parallel operations out-of-the-box—providing improved intercommunication among software library components and enhanced thread safety—are available and expanding for Intel® Integrated Performance Primitives (IPP) and Intel® Math Kernel Library (MKL). These kinds of issues are being taken care of so parallel software developers can keep pace with emerging multi-core architectures.

Enabling ISVs with Tools and Solutions

Are we seeing developer progress in these areas?
We are steadily moving in the right direction by developing improved tools and through progress in Intel's work to enable independent software vendors (ISVs). To keep the momentum behind parallel computing, we have to work with ISVs on what I call reference platform implementations—software platforms upon which developers can build differentiated applications so that the ISVs can make money and add value. The encoders and the decoders needed for video processing are a simple example of this. Organizations—from small mom-and-pop software companies with a single product to the largest Hollywood studios—may need the same capability, but it may be in different forms.

We are beginning to show the way with some reference implementations—such as with codecs optimized for parallel operation—so the development community will be able to move ahead more efficiently. A parallel game engine that is open source is a useful example of an approach that would help advance multithreaded application development.

Part of the challenge appears to be educating and supporting developers to master this technology. I'd like to send a wake-up call to the developer community and alert them: if they haven't heard about it, Intel has many different tools to offer and has a number of enabling efforts under way. Working closely with several ISVs, Intel focuses on helping them quickly prepare their software applications to be optimally threaded—designed correctly for more than four cores. Intel is also working with universities to provide enhanced training in threading in undergraduate programming courses.

We're hoping to make it clear to ISVs that they have to program in a new way to achieve performance gains on a multi-core platform. Across the board, Intel is supporting undergraduate education, providing a strong selection of very advanced tools, and engaging in direct enabling activities with ISVs. We have accomplished some very positive achievements and made substantial gains in parallel computing. We have more to do to fully realize the potential and capitalize on the opportunities.

More Info

You can discover much more about Intel's software technologies and parallel computing efforts by visiting the following areas of the Intel Web site:

Intel® Software
Intel® Threading Tools
Energy-Efficient Performance
The Evolution of Parallel Computing

David J. Kuck Bio

David J. Kuck, Intel Fellow, Software and Solutions Group Director, Parallel and Distributed Solutions Division, Intel Corporation
David J. Kuck is an Intel Fellow, Software and Solutions Group, and director of the Parallel and Distributed Solutions Division (PDSD). PDSD provides software tools and technologies to enable parallel program development for multi-core and cluster environments, and works directly with ISVs and end users to help them create high-performance parallel applications. PDSD and its Advanced Computing Center (led by Kuck) also engage in leading-edge computer system design and analysis.

Kuck was previously part of Intel's Enterprise Platforms Group and Director of the KAI Software Lab, a leading provider of performance-oriented compilers and programming tools used in the development of multithreaded applications. Kuck founded KAI in 1979 and is an emeritus faculty member of the Computer Science and Electrical and Computer Engineering departments of the University of Illinois at Urbana-Champaign. He also served as director of the Center for Supercomputing Research and Development.

Kuck holds a B.S.E.E. from the University of Michigan, as well as a master's degree and Ph.D. in engineering from Northwestern University. He is a fellow of the American Association for the Advancement of Science, the Association for Computing Machinery, and the Institute of Electrical and Electronics Engineers. He is also a member of the National Academy of Engineering. Kuck holds two patents and has published over 100 papers.

—End of Technology@Intel Magazine Article—